Input-Dynamic Distributed Algorithms for Communication Networks

Klaus-Tycho Foerster
University of Vienna

Janne H. Korhonen
IST Austria

Ami Paz
University of Vienna

Joel Rybicki
IST Austria

Stefan Schmid
University of Vienna

ABSTRACT

Consider a distributed task where the *communication network* is fixed but the *local inputs* given to the nodes of the distributed system may change over time. In this work, we explore the following question: if some of the local inputs change, can an existing solution be updated efficiently, in a dynamic and distributed manner?

To address this question, we define the *batch dynamic* CONGEST model in which we are given a bandwidth-limited communication network and a *dynamic* edge labelling defines the problem input. The task is to maintain a solution to a graph problem on the labelled graph under *batch changes*. We investigate, when a batch of α edge label changes arrive,

- how much time as a function of α we need to update an existing solution, and
- how much information the nodes have to keep in local memory between batches in order to update the solution quickly.

Our work lays the foundations for the theory of input-dynamic distributed network algorithms. We give a general picture of the complexity landscape in this model, design both universal algorithms and algorithms for concrete problems, and present a general framework for lower bounds. The diverse time complexity of our model spans from constant time, through time polynomial in α, and to α time, which we show to be enough for any task.

CCS CONCEPTS

• **Networks** → **Network algorithms**; • **Theory of computation** → **Distributed computing models**; **Dynamic graph algorithms**.

KEYWORDS

dynamic graph algorithms; congest model; distributed algorithms; communication networks; network management

ACM Reference Format:
Klaus-Tycho Foerster, Janne H. Korhonen, Ami Paz, Joel Rybicki, and Stefan Schmid. 2021. Input-Dynamic Distributed Algorithms for Communication Networks. In *Abstract Proceedings of the 2021 ACM SIGMETRICS / International Conference on Measurement and Modeling of Computer Systems (SIGMETRICS '21 Abstracts), June 14–18, 2021, Virtual Event, China.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3410220.3453923

SIGMETRICS '21 Abstracts, June 14–18, 2021, Virtual Event, China

ACM ISBN 978-1-4503-8072-0/21/06.
https://doi.org/10.1145/3410220.3453923

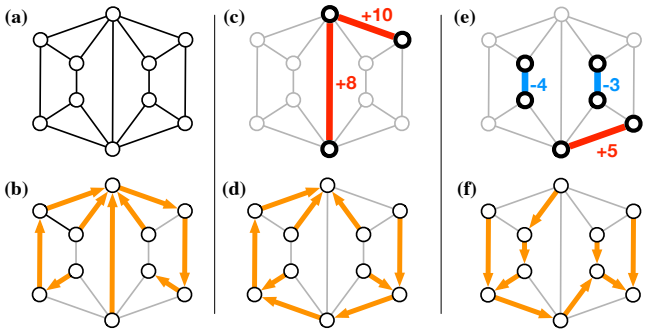

Figure 1: Examples of an input-dynamic minimum-weight spanning tree. (a) The underlying communication graph, with all edges starting with weight 1. (b) A feasible minimum-weight spanning tree. (c) A batch of two edge weight increments. (d) Solution to the new input labelling. (e) A new batch of three changes: two decrements and one increment. (f) An updated solution.

1 INTRODUCTION

Large communication networks are dynamic: network operators perform link weight updates for dynamic traffic engineering or to adjust link layer forwarding in local area networks, content distribution providers dynamically optimise cache assignments, and traffic patterns naturally evolve over time. However, while some "input parameters" change over time, the underlying wired communication topology is typically relatively static (e.g. layout and connections of the physical network equipment).

In this setting, it can be inefficient to always recompute the data structures and other information related to the operation of the network from scratch upon each change; rather, it is important to update these structures efficiently and reliably. In particular, it is desirable that if there are only few changes, then the existing solution could be efficiently utilised for computing a new solution.

Formally, many network tasks can be modelled as distributed graph problems, where we want to understand the power and limitations of dynamic distributed optimisations. To this end, we initiate the study of input-dynamic distributed graph algorithms, with the goal of laying the groundwork for a comprehensive theory.

In particular, we introduce the *batch dynamic* CONGEST model, which allows us to formally develop a theory of input-dynamic graph algorithms. This model hence complements prior work focused on how to operate in dynamic environments where the underlying communication network changes, such as temporally dynamic graphs [**?**] and distributed dynamic graph algorithms [**? ?**].

In brief, the model is a dynamic variant of the standard CONGEST model of distributed computation with following characteristics:

(1) The communication network is represented by a static graph $G = (V, E)$ on $|V| = n$ nodes. The nodes can communicate with each other over the edges, with $O(\log n)$ bandwidth per round. (This is the standard CONGEST model [**?**].)
(2) The input is given by a *dynamic* edge labelling of G. The input labelling may change and once this happens nodes need to compute a new feasible solution for the new input labelling. The labelling can denote, e.g., edge weights or mark a subgraph of G. We assume that the labels can be encoded using $O(\log n)$ bits so that communicating a label takes a single round.
(3) The goal is to design a distributed algorithm which maintains a solution to a given graph problem on the labelled graph under *batch changes*: up to α labels can change simultaneously, and the nodes should react to these changes. The nodes may maintain a *local auxiliary state* to store, e.g., the current output and auxiliary data structures, in order to facilitate efficient updates upon subsequent changes.

2 CONTRIBUTIONS

We focus on the following questions. When a batch of α edge label changes arrive, and the communication graph has diameter D,

(a) how much time does it take to update an existing solution, as a function of α and D, and
(b) how much information does a node need to keep in its local memory between batches, to achieve optimal running time?

With these questions, we lay the foundations for the theory of input-dynamic distributed graph algorithms. We draw a general picture of the complexity landscape in the batch dynamic CONGEST model as summarised in Table 1. Our main results are as follows.

Universal upper bounds. As an almost trivial baseline, we observe that *any* graph problem can be solved in $O(\alpha + D)$ rounds. Moreover, any graph problem where the output of a node depends only on the constant-radius neighbourhood of the node – solvable in $O(1)$ rounds in the LOCAL model[1] – can be solved in $O(\alpha)$ rounds. However, these universal algorithms come at a large cost in space complexity: storing the auxiliary state between batches may require up to $O(m \log n)$ bits, where m is the number of edges — in the input graph if the input marks a subgraph, and in the communication graph if the input represents edge weights.

Intermediate complexity: clique enumeration. We give an algorithm for enumerating k-cliques in $O(\alpha^{1/2})$ rounds, beating the universal upper bound for local problems, and showing that there exist non-trivial problems that can be solved in $o(\alpha)$ rounds. To complement this result, we show that dynamic clique detection requires $\Omega(\alpha^{1/4})$ rounds. This is an example of a natural problem with time complexity that is neither constant nor $\Theta(\alpha)$.

Saving space: minimum-weight spanning trees. We show that a minimum-weight spanning tree[2] can be maintained in $O(\alpha + D)$ rounds using only $O(\log n)$ bits per node for storing the auxiliary state; this exponentially improves the storage requirements of a previous distributed dynamic algorithm of Peleg [**?**], which uses $O(n \log n)$ bits of memory per node. In addition, we show that our result is tight, in terms of update time, up to $\operatorname{poly}\log \alpha$: for any $\alpha \le n^{1/2}$, maintaining a minimum-weight spanning tree requires $\Omega(\alpha/\log^2 \alpha + D)$ rounds.

[1] The LOCAL model is similar to the CONGEST model, but without the $O(\log n)$ limitation on the message sizes [**?**].
[2] See Fig. 1 for an example.

Table 1: Upper and lower bounds for selected problems in batch dynamic CONGEST. Upper bounds marked with † follow from the universal algorithms. The lower bounds apply in a regime where α is sufficiently small compared to n.

	Upper bound		Lower bound
Problem	Time	Space	Time
any problem	$O(\alpha + D)$	$O(m \log n)$	—
any LOCAL(1) probl.	$O(\alpha)$	$O(m \log n)$	—
min. spanning tree	$O(\alpha + D)$	$O(\log n)$	$\Omega(\alpha/\log^2 \alpha + D)$
k-clique	$O(\alpha^{1/2})$	$O(m \log n)$	$\Omega(\alpha^{1/4}/\log \alpha)$
4-cycle	$O(\alpha)^\dagger$	$O(m \log n)^\dagger$	$\Omega(\alpha^{2/3}/\log \alpha)$
k-cycle, $k \ge 5$	$O(\alpha)^\dagger$	$O(m \log n)^\dagger$	$\Omega(\alpha^{1/2}/\log \alpha)$
diam., $(3/2 - \varepsilon)$-apx.	$O(\alpha + D)^\dagger$	$O(m \log n)^\dagger$	$\Omega(\alpha/\log^2 \alpha + D)$
APSP, $(3/2 - \varepsilon)$-apx.	$O(\alpha + D)^\dagger$	$O(m \log n)^\dagger$	$\Omega(\alpha/\log^2 \alpha + D)$

A general framework for lower bounds. We develop a framework for lifting CONGEST lower bounds into the batch dynamic CONGEST model, providing a vast array of non-trivial lower bounds for input-dynamic problems. These include lower bounds for classic graph problems, such as cycle detection, clique detection, computing the diameter, approximating all-pairs shortest paths, and computing minimum spanning trees. The lower bounds hold for both deterministic and randomised algorithms.

Dynamic congested clique. We explore the dynamic variant of the *congested clique* model, which arises as a natural special case of the batch dynamic CONGEST. We show that triangle counting can be solved in $O((\alpha/n)^{1/3} + 1)$ rounds in this model using $O(n \log n)$ bits of auxiliary state by applying a *dynamic matrix multiplication* algorithm. To contrast this, we show that any problem can be solved in $O(\lceil \alpha/n \rceil)$ rounds using $O(m \log n)$ bits of auxiliary state.

ACKNOWLEDGMENTS

We thank Jukka Suomela for discussions. Research supported by the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement No 805223 ScaleML), from the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska–Curie grant agreement No. 840605, from the Vienna Science and Technology Fund (WWTF) project WHATIF, ICT19-045, 2020-2024, and from the Austrian Science Fund (FWF) and netIDEE SCIENCE project P 33775-N.

REFERENCES

[] Baruch Awerbuch, Israel Cidon, and Shay Kutten. 2008. Optimal Maintenance of a Spanning Tree. *J. ACM* 55, 4 (Sept. 2008).
[] Arnaud Casteigts, Paola Flocchini, Walter Quattrociocchi, and Nicola Santoro. 2012. Time-varying graphs and dynamic networks. *IJPEDS* 27, 5 (2012), 387–408.
[] Keren Censor-Hillel, Elad Haramaty, and Zohar S. Karnin. 2016. Optimal Dynamic Distributed MIS. In *Proc. PODC*. 217–226.
[] David Peleg. 1998. Distributed matroid basis completion via elimination upcast and distributed correction of minimum-weight spanning trees. In *Proc. ICALP*. Springer, 164–175.
[] David Peleg. 2000. *Distributed Computing: A Locality-Sensitive Approach.* Society for Industrial and Applied Mathematics.